Modeling Priority-based Incentive Policies for Peer-assisted Content Delivery Systems*

Niklas Carlsson and Derek L. Eager

Department of Computer Science, University of Saskatchewan
Saskatoon, SK S7N 5C9, Canada
carlsson@cs.usask.ca, eager@cs.usask.ca

Abstract. Content delivery providers can improve their service scalability and offload their servers by making use of content transfers among their clients. To provide peers with incentive to transfer data to other peers, protocols such as BitTorrent typically employ a tit-for-tat policy in which peers give upload preference to peers that provide the highest upload rate to them. However, the tit-for-tat policy does not provide any incentive for a peer to stay in the system beyond completion of its download.

This paper presents a simple fixed-point analytic model of a priority-based incentive mechanism which provides peers with strong incentive to contribute upload bandwidth beyond their own download completion. Priority is obtained based on a peer's prior contribution to the system. Using a two-class model, we show that priority-based policies can significantly improve average download times, and that there exists a significant region of the parameter space in which both high-priority and low-priority peers experience improved performance compared to with the pure tit-for-tat approach. Our results are supported using event-based simulations.

Keywords: Modeling, peer-assisted content delivery, priority-based incentive

1 Introduction

A highly scalable approach to content delivery is to utilize the upload bandwidth of the clients to offload the original content source [1, 2, 3, 4, 5, 6]. Existing peer-to-peer download protocols, such as BitTorrent [1], allow each peer to download content from any peer that has content that it does not have, and do not require any organized delivery structure. With these protocols each file is split into many small pieces, each of which may be downloaded from different peers.

To provide peers with an incentive to upload pieces to other peers, these protocols typically employ a tit-for-tat policy in which peers give upload preference to peers that provide the highest upload rate to them. Whereas such policies provide peers with strong incentives to upload at their full upload capacity while downloading a file,

---

they do not provide peers with any incentive to serve additional data after having completed their download. Therefore, unless altruistic peers graciously continue to serve data beyond their download completion, peer-assisted systems with limited server resources are not able to provide download rates much higher than the peer upload rate. When the peer download capacity significantly exceeds the upload capacity, as is commonly the case with home Internet connections (e.g., [7]), much of the download capacity in such systems is therefore underutilized.

To increase the available upload resources some BitTorrent sharing communities enforce a minimum required sharing ratio (defined as the ratio between the total amount of data uploaded and downloaded); other communities rely on social etiquette (e.g., a sharing ratio may be advertised without any enforcement mechanism). While improved performance has been observed in such communities [8, 9], no policies appear to be in place to reward peers contributing more than the advertised minimum.

This paper presents a simple fixed-point model of a two-class, priority-based incentive mechanism which uses differentiated service to provide peers with a strong incentive to upload data past their download completion. In addition to using a tit-for-tat policy to encourage peers to upload as much as possible, while downloading, a fraction of the upload bandwidth of each peer/server is used to serve high-priority class peers. Priority is obtained based on a peer's prior contribution to the system.

We show that this policy is able to significantly improve average download times and provides peers with strong incentive to contribute past their download completion. Furthermore, there exists a significant region of the parameter space in which the performance of both high-priority and low-priority peers is improved over that with the pure tit-for-tat approach. We also discuss the impact of individual client choices on system performance. Results are supported using event-based simulations.

Previous multi-class models of BitTorrent-like systems assume that the arrival rate of each class of peers is known and each peer allocates a static portion of upload bandwidth to each class of peers [10]. In contrast, we consider scenarios in which the fraction of upload bandwidth used for uploads to peers of each class depends on the proportion of peers of each class, which in turn depends on the extent to which clients decide to gain high-priority status through their contribution of upload resources. Naturally, these decisions depend on client perception of the advantage of being a high-priority peer versus the additional cost. In this work a fixed-point is determined such that the performance differential enjoyed by high-priority peers for a particular proportion of peers, provides incentive that yields exactly that proportion of peers choosing to earn high-priority status by contributing as a seed. To achieve good accuracy over the considered portion of the system parameter space, we also require a somewhat more detailed mean value analysis than previous fluid models [10, 11, 12, 13]. These differences are further discussed in Section 4.

Other related work has included the use of analytic models [14, 15] or instrumentation [16, 17, 18] to capture the peer interaction, the study of reputation-based (e.g., [19]) and currency/token-based (e.g., [20]) reputation systems, and the use of taxation techniques when rewarding peers [21].

The remainder of the paper is organized as follows. Section 2 provides a brief overview of BitTorrent. A simple priority-based policy is presented in Section 3. Section 4 presents our analytic model. Results and validation are presented in Section 5. Finally, conclusions are presented in Section 6.

2 BitTorrent Overview

We consider here the use of BitTorrent-like protocols to aid in the on-demand delivery of files stored on one or more content provider servers. BitTorrent [1] splits files into *pieces*, which themselves are split into smaller *sub-pieces*. Multiple sub-pieces can be downloaded in parallel from different peers. A peer is said to *have* a piece whenever the entire piece is downloaded, and is *interested* in all peers that have at least one piece that it currently does not have itself. Peers that have the entire file are called *seeds*, while peers that only have parts of the file and are still downloading are called *leechers*. Each peer establishes persistent connections with a large set of peers; however, at each time instance each peer only uploads to a limited number of peers (these peers are considered *unchoked*, while all other peers are *choked*). A rate-based *tit-for-tat* policy is used to encourage peers to upload pieces and discourage free-riding. With this policy leechers give upload preference to the leechers that provide the highest download rates. To probe for better pairings (or in the case of the seeds, allow new peers to download pieces), periodically each peer uses an optimistic unchoke policy, in which a random peer is unchoked. To ensure high piece diversity a *rarest-first* piece selection policy is used to determine which pieces to request next.

3 A Priority-based Incentive Policy

This section presents a priority-based incentive policy with two priority classes. We consider a system from which clients may download many files. New clients that have yet to contribute upload bandwidth as a seed are (by default) given low priority. Each time a peer uploads a (combined) total of L_H data as a seed (over some number of downloaded files) it earns high-priority status for one future file download. In return for their extra upload contributions, high-priority peers receive better service. Service differentiation is achieved by requiring that every peer/server use a fraction f_H of their upload bandwidth to serve high-priority peers. The remaining upload bandwidth is allocated based on tit-for-tat with optimistic unchoking. This policy provides peers with incentives to both upload at a high rate when downloading (from use of tit-for-tat), and to upload additional data as a seed (from use of priority).[1]

When selecting new peers to upload to, a peer randomly selects a high-priority peer if at least one such peer is requesting service and less than a fraction f_H of the data it has uploaded has been uploaded using priority uploading; otherwise, tit-for-tat with optimistic unchoking is used. To encourage bidirectional tit-for-tat relationships, in which connections are responsive to the tit-for-tat policy (with optimistic unchoking), the unchoke policy considers the download rates of priority connections to be zero.

To ensure that peers allocate roughly a fraction f_H of their upload bandwidth to priority uploading, when possible, each peer keeps track of the amount of data uploaded using (i) tit-for-tat with optimistic unchoking, and (ii) priority uploading.

---

[1] Many policy variations are possible, including variants in which service differentiation is achieved with smoothly varying weights rather than with a binary priority assignment.

When the upload of a new piece begins, the amount credited to the corresponding upload category is increased by the size of a piece. This credit may later be reduced, however, if due to parallel downloading, the actual amount uploaded by the peer is less than the size of the piece. Upload amounts are credited only during periods when there is at least one high-priority peer to which an upload could occur.

We assume that the system provides all clients with information about the average download times of high-priority and low-priority clients, as well as the amount of data that must be uploaded as a seed to become high priority. This information (provided by a tracker, for example) allows clients to make an informed decision as to whether to make the upload contribution required for high-priority status.

4 Model Description

This section presents our analytic model. We assume that all servers and peers have implemented the priority-based incentive policy described in Section 3 and are equipped with software (provided by the content provider, for example) which monitors the peers' upload contributions. While such software cannot control the upload bandwidth made available by peers, we assume that the software allows the fraction f_H used for priority uploading to be maintained and peers to be fairly and correctly labeled as either high or low priority.[2]

It is further assumed that the system operates in steady state and clients make many requests to the system. Furthermore, we assume that the probability a peer decides to upload a particular amount of data as a seed (so as to gain credit towards high priority status), after a file download, is independent of file identity, and that no peer accumulates high-priority download credit that it does not use in later downloads. Thus, in steady state, the total average rate that seeds upload data for any particular file will equal L_H times the request rate of high-priority peers for that file. We assume that peers do not depart the system before having fully downloaded the file.

Each client is assumed to determine the extent to which it is willing to serve as a seed according to the performance difference (as measured by the server) between the average download times of high-priority and low-priority peers. We analyze here a simple case where each peer has a threshold such that the peer will serve exactly L_H data as a seed following each download, should this performance difference exceed the threshold. Assuming that these thresholds follow some (arbitrary) distribution, our analysis obtains a fixed-point solution such that the performance differential enjoyed by high-priority peers for a particular proportion of peers, yields exactly that proportion of peers choosing to earn high-priority status by contributing as a seed.

We model download performance for a single representative file of size L. It is assumed that clients have an average upload bandwidth capacity U, average download bandwidth capacity D, generate requests for the file that arrive according to a Poisson

---

[2] A reporting and tracking system could be used to identify misbehaving peers that do not allocate the required fraction of upload bandwidth to priority uploading. Such peers could be punished by black-listing them from receiving uploads by servers and well-behaved peers.

process with rate λ, and that the server(s) have a total available upload bandwidth for the file equal to s times the average peer upload bandwidth U. Subscripts H and L are used to distinguish quantities for the high-priority and low-priority class, respectively.

4.1 General Model

Given the above assumptions, a differential equation expressing the relationship between the average achieved download rate d_c and the average number x_c of clients actively downloading the file at any given time t, for each class c, can be derived as

$$dx_c / dt = \lambda_c - x_c d_c / L, \quad c = L, H \tag{1}$$

where d_c/L is the rate at which peers of class c complete their downloads and λ_c is the rate at which peers of class c begin new downloads. Focusing on the steady state case (i.e., when $dx_c/dt = 0$), this yields

$$x_c = \lambda_c (L / d_c). \quad c = L, H \tag{2}$$

We note that these expressions can easily be obtained directly using Little's law. Similarly, the relationship between the average achieved upload rate u_y of peers that are currently acting as seeds, and the average number of such peers, is given by

$$y = \lambda_H (L_H / u_y). \tag{3}$$

To continue the analysis so as to achieve acceptable accuracy over the considered portion of the parameter space, we require a more detailed mean value analysis than previous fluid models [10, 11, 12]. Our model uses conditional expected values (rather than only the long term averages) and reallocates upload bandwidth intended to be used for high-priority peers, but not allocated owing to insufficient total download capacity of these peers, to low-priority peers. Let $x_{c(Q)}$ and $y_{(Q)}$ denote the expected number of class c leechers, and the expected number of seeds, respectively, conditioned on the set of constraints Q. For example, if Q implies that $x_c = a_c$, then $x_{c(Q)} = a_c$; similarly, if Q implies that $x_c \geq 1$, then $x_{c(Q)} = x_c /(1 - e^{-x_c})$. Further, let $d_{c(Q)}$ and $N_{(Q)}$ denote the corresponding conditional expected values for the download rate and the number of peers and servers uploading data, respectively, where the servers are considered as s peers (since their total upload bandwidth is sU). The download and upload rates of each class of clients can then be expressed as follows:

$$d_H = d_{H(x_H \geq 1)}; d_L = d_{L(x_L \geq 1)}; u_y = (x_H d_{H(x_H \geq 1, y \geq 1)} + x_L d_{L(x_L \geq 1, y \geq 1)}) / N_{(y \geq 1)}. \tag{4}$$

These expressions make use of the fact that the download and upload rate of a peer type is only applicable if there is at least one peer of that particular type in the system.

In determining the conditional download rates, we separate upload bandwidth used for priority uploading (shared only among high-priority leechers) and tit-for-tat and optimistic unchoking (shared among all leechers). This separation allows the download rates to be adjusted based on the relative mix of peers of each class. Given

a total upload capacity of $N_{(Q)}U$, the download rate of high-priority peers is given by

$$d_{H(Q)} = \min\left[D, N_{(Q)}U\left(\frac{1 - f_H}{x_{H(Q)} + x_{L(Q)}} + \frac{f_H}{x_{H(Q)}} \right) \right].$$ (5)

With all the remaining bandwidth available to them (after high-priority peers are satisfied) the download rate of the low-priority peers can be expressed as follows:

$$d_{L(Q)} = \min\left[D, \frac{N_{(Q)}U - x_H d_{H(Q)}}{x_{L(Q)}} \right].$$ (6)

This expression reduces to $N_{(Q)}U(1 - f_H)/(x_{H(Q)} + x_{L(Q)})$ whenever $d_{H(Q)} \leq D$. Finally, the expected number of peers or servers effectively uploading data is given by

$$N_{(Q)} = s + y_{(Q)} + \sum_{c=L,H} x_{c(Q)} - \Pr(x_H + x_L = 1|Q),$$ (7)

where $\Pr(x_H + x_L = 1 \mid Q)$ is the probability that there only is a single leecher in the system. This expression assumes that every peer can upload data whenever there is more than one client in the system. Assuming the number of high-priority and low-priority peers is independent given Q, the above probability can be calculated as

$$\Pr(x_H + x_L = 1|Q) = \Pr(x_H = 1|Q)\Pr(x_L = 0|Q) + \Pr(x_H = 0|Q)\Pr(x_L = 1|Q),$$ (8)

where the conditional probabilities can be calculated using standard theory.

Given the portion λ_H/λ of peers selecting to become high-priority peers, as would depend on the relative performance difference ($T_L/T_H - 1$), the average download rates d_c and download times $T_c = L/d_c$ can be easily obtained by numerically solving the above equation system. As with other fluid models, the complexity of these calculations is independent of the number of peers in the system.

4.2   All High-priority Case

This section describes how our model can be reduced to a closed form solution for the special case in which (i) all clients select to become high-priority peers, (i) on average serve L_H data as a seed, (iii) the system is demanding $\lambda L \gg d$, and (iv) the seed contributions are significant $\lambda L_H \gg u_y$.

For this special case the conditional values are roughly equal to the long term averages and equations (4) and (5) reduce to $d_H = \min[D, \dfrac{s + \lambda L / d_H + \lambda L_H / u_y}{\lambda L / d_H} U]$,

where $u_y = \lambda L/(s + \lambda L/d_H + \lambda L_H/u_y)$. Here Little's law has been used to substitute the average number of leechers $x_H = \lambda L/d_H$ and seeds $y = \lambda L_H/u_y$, respectively. Solving for d_H allows an expression for the average download time $T_H = L/d_H$ to be obtained:

$$T_H = \min\left[L/D, (L - L_H)/U - s/\lambda \right].$$ (9)

Note that as long as $D \geq U/(1 - sU/(\lambda L))$, the download time decreases linearly with the amount of data L_H uploaded by seeds, and the clients fully utilize their maximum download bandwidth capacity D when seeds upload $L_H^* = L(1 - U/D) - sU/\lambda$ data. While equation (9) also is applicable to a regular tit-for-tat system in which altruistic peers serve L_H data as seeds, we note that such seed contributions are much more likely in the priority system, in which the peers are given a performance incentive.

5 Results

This section validates the model and analyzes the characteristics of the priority-based policy. For validation we modified an existing event-based simulator of BitTorrent-like systems [22]. It is assumed that no connections are choked in the middle of an upload, and peers only request new downloads when they are not fully utilizing their download capacity D. For simulating the rate at which pieces are exchanged, it is assumed that connection bottlenecks are located at the end points (i.e., either by the upload capacity U at the sender or by the download capacity D at the receiver) and the network operates using max-min fair bandwidth sharing (using TCP, for example).

5.1 Validation using Known User Choices

In this section we validate the analytic model for the case where the proportion of clients of each priority class is known. We consider a system with a single server operating in steady state. The system is simulated for 6400 requests, with the initial 1000 and the last 400 requests removed from the measurements. Each data point represents the average of 5 simulations. The file is split into 512 pieces. The peers and the server can simultaneously upload to at most 4 and $4s$ peers, respectively.

Without loss of generality, the file size L and upload bandwidth capacity U are fixed at one. With these normalized units, the volume of data transferred is measured in units of the file size and the download time is measured in units of the minimum time it takes for a client to fully upload a copy of the file.

Fig. 1 presents the average download times for each priority class individually and the overall averages, as predicated by both the analytic model and simulations. For each sub-figure in Fig. 1, one variable is considered at a time. The default case assumes that $\lambda = 100$, $D/U = 3$, $L_H/L = 20\%$, $s = 1$, $f_H = 25\%$, and $\lambda_H/\lambda = 50\%$. Fig. 1(a) also presents the results for a probabilistic model extension.[3] For the other sub-figures the difference between the general model and the probabilistic extension is not noticeable and the results for the extension are therefore omitted.

---

[3] Due to space constraints the description of this extension is omitted. The basic idea is to split each condition Q used in expression (4) into three sub-cases: (i) there is no additional leecher in the system, (ii) there is at least one high-priority leecher in the system, and (iii) there is no high-priority leecher in the system, but at least one low-priority leecher.
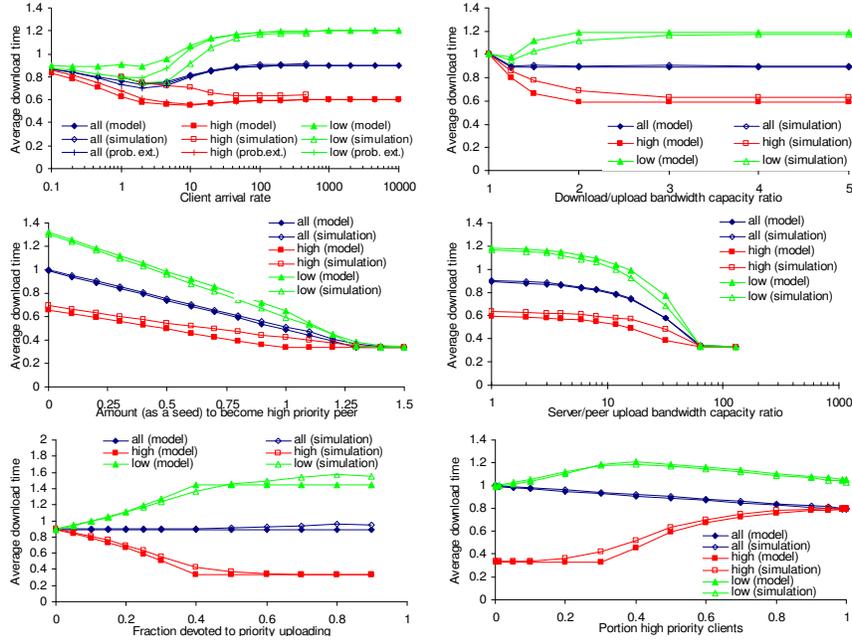
Fig. 1. Model validation. Default settings: $L = 1$, $U = 1$, $\lambda = 100$, $D/U = 3$, $L_H = 0.2$, $s = 1$, $f_H = 0.25$, $\lambda_H/\lambda = 0.5$. (a) Client arrival rate λ. (b) Download/upload capacity ratio D/U. (c) Amount L_H to be uploaded as a seed. (d) Server/peer upload capacity ratio s. (e) Fraction f_H devoted to priority uploading. (f) Portion λ_H/λ high priority peers.

The agreement between simulation and analytical results is excellent for the overall averages. The agreement for the averages for each individual class is good, although the discrimination between the classes is typically slightly smaller for the simulations. This difference is likely due to the fixed-point model not capturing variability in the number of active peers of each class. Note that larger errors are observed in regions with larger variability; for example, in systems with relatively low arrival rate. For these regions, the probabilistic extension, which takes more state information into account, provides a slightly better approximation. In general, however, we have found that the added complexity of such models is not typically justified.

As suggested by the closed form equations in Section 4.2, the average download times decrease linearly with the amount of content uploaded by seeds (with the slope determined by the portion of high-priority peers in the system). The discrimination between the priority classes increases linearly with the fraction devoted to priority uploading. We note that the decreased performance of low-priority peers (relative to the high-priority peers) is offset by the overall improved performance due to high priority peers making additional upload contributions as seeds. As further discussed in Section 5.2, such contributions can in some cases allow the low-priority peers to achieve better download times than peers in non-priority systems.
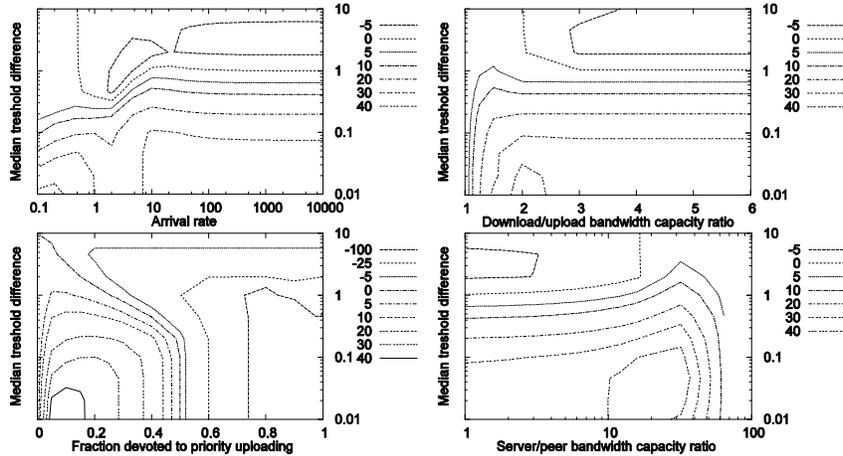
Fig. 2. Percentage decrease in download times for low-priority peers when the clients individual thresholds follow an exponential distribution. Default settings are: $L = 1$, $U = 1$, $L_H = 0.5$, $\lambda = 100$, $D/U = 3$, $s = 1$, $f_H = 0.25$.

5.2 Social Welfare Regions

In this section we consider under what circumstances low-priority peers experience download times no worse (or only slightly worse) than they would in a regular tit-for-tat system. We call regions in the modeling space for which both classes of peers see improved performance social welfare regions.

We consider a system without altruistic peers and model the performance of a regular tit-for-tat system using the special case in which $L_H = 0$ and $f_H = 0$. Assuming that any additional (unconditional) upload contributions made by altruistic peers are equal for both systems, the results for systems with altruistic peers correspond to scenarios with additional server bandwidth.

Fig. 2 shows the contour lines for different levels of improvements observed by the low-priority peers. These experiments assume that the clients' individual thresholds follow an exponential distribution. On the y-axis of each sub-figure the median of this distribution is plotted. For example, with a median of 1, 50% of the clients require the performance difference to be at least 100% before selecting to become a high-priority peer. With our analytic model providing slight overestimates of the download times of the low-priority peers, we expect that in real systems the social regions would likely be somewhat larger than indicated here.

Note that the size of the social welfare regions typically increases as clients are more willing to serve as high-priority peers (i.e., for lower thresholds). In fact, for large portions of the parameter space the low-priority peers actually observe significant performance improvements. Only when clients have large thresholds or the fraction of bandwidth devoted to priority uploading exceeds 50%, do the low-
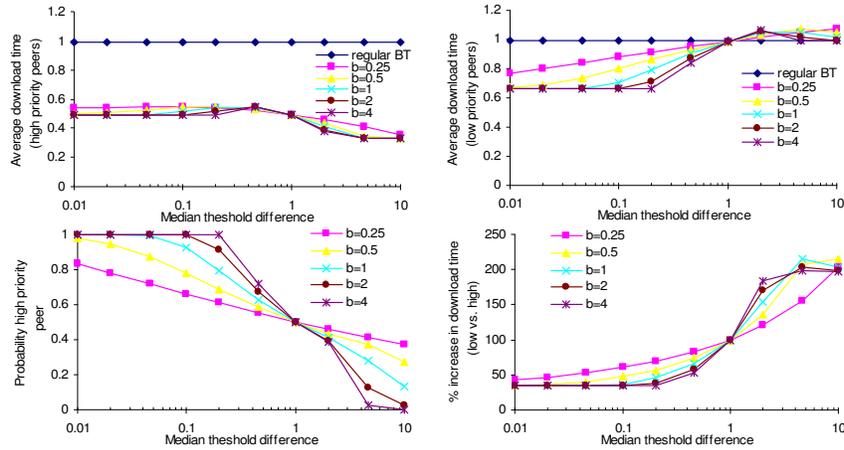
Fig. 3. Performance impact of the shape parameter β and the overall aggressiveness of clients. Default settings are: $L = 1$, $U = 1$, $L_H = 0.5$, $\lambda = 100$, $D/U = 3$, $s = 1$, $f_H = 0.25$.

priority peers observe decreased performance. It should further be noted that with $f_H = 0.25$ and $L_H = 0.5$ the low-priority peers do not have more than a 10% increase in download times (independent of the peer threshold distribution). The corresponding values when $L_H = 0.25$ and $L_H = 0.75$ are 25% and 0%, respectively.

5.3 User Behavior and System Dynamics

To illustrate the impact that individual client choices have on the performance of the system dynamics, Fig. 3 shows the performance using different threshold distributions. To cover a wide range of cases we assume that the thresholds follow a Weibull distribution. The shape parameter β determines if the distribution is light-tailed (β ≥ 1), or heavy-tailed (β < 1); with β = 1 corresponding to an exponential distribution and β = 3.4 approximating a normal distribution. Note that the curves become more and more step-like as the shape parameter grows and the distribution becomes more compressed (with, in the limiting case, all peers sharing the same threshold). All curves cross at the point where the portion of high-priority peers is 50%; this is since the distributions are normalized to have the same median and the fact that the system performance is mainly determined by the fraction of peers that have a threshold below some "critical" value (not the shape of the threshold distribution itself). Another observation from these figures is that low-priority peers only perform worse (and only slightly worse) than peers in a regular BitTorrent system when the median threshold difference is relatively large (e.g., 100%).
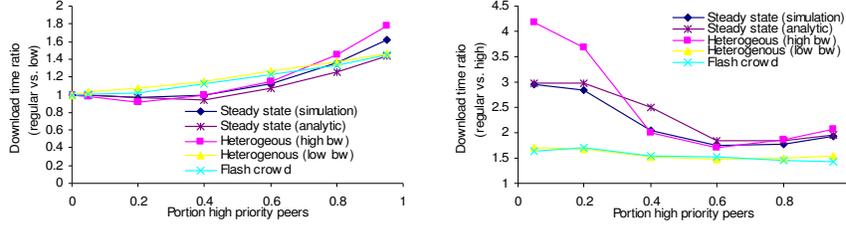
Fig. 4. Example scenarios: (i) steady state ($\lambda = 100$), (ii) flash crowd ($\lambda_0 e^{-\gamma t}$, with $\lambda_0 = 791.0$ and $\gamma = 1$), and (iii) heterogeneous scenario ($\lambda = 50$, $\lambda_A/\lambda = 0.5$, $U_A = 0.2$, $D_A = 0.6$, $U_B = 1$, $D_B = 3$). Default settings are: $L = 1$, $U = 1$, $L_H = 0.5$, $D/U = 3$, $s = 1$, $f_H = 0.25$.

5.4 Alternative Scenarios and Workload Assumptions

This section considers alternative scenarios and workload assumptions than those assumed by our analytic model. First, motivated by measurement studies [12], we consider a flash-crowd scenario in which peers arrive at an exponentially decaying rate $\lambda(t) = \lambda_0 e^{-\gamma t}$, with $\lambda_0 = 791.0$ and $\gamma = 1$. For this case 63.2% of all 500 peer arrivals occur within the first time unit. Second, a steady state scenario is considered with heterogeneous peers: low-bandwidth ($U_A = 0.2$, $D_A = 0.6$) and high-bandwidth ($U_B = 1$, $D_B = 3$) peers. For reference we also include our original default scenario.

Fig. 4 shows the relative download times for each scenario. Although the performance improvements are smaller in the flash crowd scenario and for the low-bandwidth clients in the heterogeneous scenario, the increased performance differences between high-priority and low-priority peers when the portion of high-priority peers are small gives a strong incentive to contribute additional resources, thereby allowing the overall performance to be improved. Again, we note that the low-priority peers do not perform much worse than peers in a regular tit-for-tat system and in many cases significantly better.

6 Conclusions

This paper presents a simple analytic model of a two-class priority-based incentive mechanism in which priority is obtained based on peers' prior contributions to the system. It is shown that priority-based policies can provide peers with strong incentive to contribute upload bandwidth beyond their own download completion, can significantly improve the average download times in the system, and that there exists a significant region in the parameter space in which both high-priority and low-priority peers experience improved performance compared to the pure tit-for-tat approach. While this paper has focused on download, we note that these types of priority policies also can be used for peer-assisted streaming [22].

References

1. Cohen, B.: Incentives Build Robustness in BitTorrent. In: Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA. (2003)
2. Gkantsidis, C., Rodriguez, P. R.: Network Coding for Large Scale Content Distribution. In: IEEE INFOCOM, Miami, FL. (2005)
3. Sherwood, R., Braud, R., Bhattacharjee, B.: Slurpie: A Cooperative Bulk Data Transfer Protocol. In: IEEE INFOCOM, Hong Kong, China. (2004)
4. Chu, Y., Rao, S. G., Zhang, H.: A Case for End System Multicast. In: ACM SIGMETRICS, Santa Clara, CA. (2000)
5. Kozic, D., Rodriguez, A., Albrecht, J., Vahdat, A.: Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh. In: ACM SOSP, Bolton Landing, NY. (2003)
6. Zhang, X., Liu, J., Li, B., Yum, T-S. P.: CoolStreaming/DONet: A Datadriven Overlay Network for Peer-to-Peer Live Media Streaming. In: IEEE INFOCOM, Miami, FL. (2005)
7. Saroiu, S., Gummadi, K. P., Gribble, S. D.: A Measurement Study of Peer-to-Peer File Sharing Systems. In: IS&T/SPIE MMCN, San Jose, CA. (2002)
8. Andrade, N., Mowbray, M., Lima, A., Wagner, G., Ripeanu, M.: Influences on Cooperation in BitTorrent Communities. In: ACM SIGCOMM Workshop on Economics of P2P Systems, Philadelphia, PA. (2005)
9. Ripeanu, M., Mowbray, M., Andrade, N., Lima, A.: Gifting Technologies: A BitTorrent Case Study. First Monday 11, 11. (2006)
10. Clévenot-Perronnin, F., Nain, P., Ross, K. W.: Multiclass P2P Networks: Static Resource Allocation for Service Differentiation and Bandwidth Diversity. In: IFIP PERFORMANCE, Juan-les-Pins, France. (2005)
11. Qiu, D., Srikant, R.: Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks. In: ACM SIGCOMM, Portland, OR. (2004)
12. Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., Zhang, X.: Measurement, Analysis, and Modeling of BitTorrent-like Systems. In: ACM IMC, Berkley, CA. (2005)
13. Kumar, R., Liu, Y., Ross, K.: Stochastic Fluid Theory for P2P Streaming Systems. In: IEEE INFOCOM, Anchorage, AK. (2007)
14. Massoulie, L., Vojnovic, M.: Coupon Replication Systems. In: Proc. ACM SIGMETRICS, Banff, Canada. (2005)
15. Lin, M., Fan, B., Chiu, D., Lui, J. C. S.: Stochastic Analysis of File Swarming Systems. In: IFIP PERFORMANCE, Cologne, Germany. (2007)
16. Legout, A., Urvoy-Keller, G., Michiardi, P.: Rarest First and Choke Algorithms Are Enough. In: ACM IMC, Rio de Janeiro, Brazil. (2006)
17. Legout, A., Liogkas, N., Kohler, E., Zhang, L.: Clustering and Sharing Incentives in BitTorrent Systems. In: ACM SIGMETRICS, San Diego, CA. (2007)
18. Piatek, M., Isdal, T., Anderson, T., Krishnamurthy, A., Venkataramani, A.: Do Incentives Build Robustness in BitTorrent?. In: NSDI, Cambridge, MA. (2007)
19. Kamvar, S. D., Schlosser, M. T., Garcia-Molina, H.: The Eigentrust Algorithm for Reputation Management in P2P Networks. In: WWW, Budapest, Hungary. (2003)
20. Zhong, S., Chen, J., Yang, Y. R.: Sprite: a Simple, Cheat-Proof, Credit-Based System for Mobile Ad-hoc Networks. In: IEEE INFOCOM, San Francisco, CA. (2003)
21. Chu, Y., Chuang, J., Zhang, H.: A Case for Taxation in Peer-to-Peer Streaming Broadcast. In: ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems, Portland, OR. (2004)
22. Carlsson, N., Eager, D. L.: Peer-assisted On-demand Streaming of Stored Media using BitTorrent-like Protocols. In: IFIP/TC6 NETWORKING, Atlanta, GA. (2007)